Cache算法设计与实现

1. **总体思路**

在本次实验要求中，给出了Cache大小（256KB），Block大小（16B），内存地址的位数为64位（数据从外部文件中逐行读取），Cache的替换算法为LRU算法，需要分别使用直接映射DM、组关联SA方法、MRU预测方法和Multi-column预测方法模拟Cache行为，并且计算对应的命中率。在本算法设计中，MRU和Multi-column都属于提高Cache的First Hit，加快Cache命中的预测算法，最终的Cache替换策略都为LRU，因此并不会改变Cache总的命中率。

本次实验使用C++语言在VSCODE编译环境中进行算法实现，主要设计思路如图1所示：

图1 Cache算法设计总体思路

主程序（Main Process）负责从外部读取TRACE文件，解析地址中的Tag、Set与Offset, 并与Cache中的组号进行映射（b = B mod C，其中b为cache中块号，B为主存中的块号，C为cache中块容量），然后选择调用LRU、MRU或Multi-column算法对CACHE进行访问或替换，同时将命中结果输出到终端，将Miss的记录保存到外部文件中。

LRU、MRU和Multi-column算法可以封装成为一个类（CloveCache类），用于对一块地址空间进行访问策略和换入换出管理。类的底层使用LinkList、Map、Vector等数据结构进行组织，便于完成不同的功能。

Cache使用数组进行模拟。每个数组单元代表Cache中的一个组，组内采用类（CloveCache类）进行管理。CloveCache类可以完成Cache组内数据的MRU预测访问、 Multi-column预测访问以及LRU换入换出等管理工作。

1. **算法设计**
2. Cache设计
3. LRU算法
4. MRU算法
5. Multi-column算法
6. **实验结果**
7. **分析总结**